Lab Task  
1. Write a program to traverse Graph through BFS using classes in C++

#include <iostream>

#include <list>

using namespace std;

class Graph {

  int numVertices;

  list<int>\* Lists;

  bool\* visited;

   public:

  Graph(int vertices);

  void edge\_add(int src, int dest);

  void BFS\_disp(int Vertex);

};

Graph::Graph(int vertices) {

  numVertices = vertices;

  Lists = new list<int>[vertices];

}

void Graph::edge\_add(int src, int dest) {

  Lists[src].push\_back(dest);

  Lists[dest].push\_back(src);

}

void Graph::BFS\_disp(int startVertex) {

  visited = new bool[numVertices];

  for (int i = 0; i < numVertices; i++)

    visited[i] = false;

  list<int> queue;

  visited[startVertex] = true;

  queue.push\_back(startVertex);

  list<int>::iterator i;

  while (!queue.empty()) {

    int curr = queue.front();

    cout << curr << " ";

    queue.pop\_front();

    for (i = Lists[curr].begin(); i != Lists[curr].end(); ++i) {

      int adjVertex = \*i;

      if (!visited[adjVertex]) {

        visited[adjVertex] = true;

        queue.push\_back(adjVertex);

      }

    }

  }

}

int main() {

  Graph g(4);

  g.edge\_add(0, 1);

  g.edge\_add(0, 2);

  g.edge\_add(1, 3);

  g.edge\_add(2, 2);

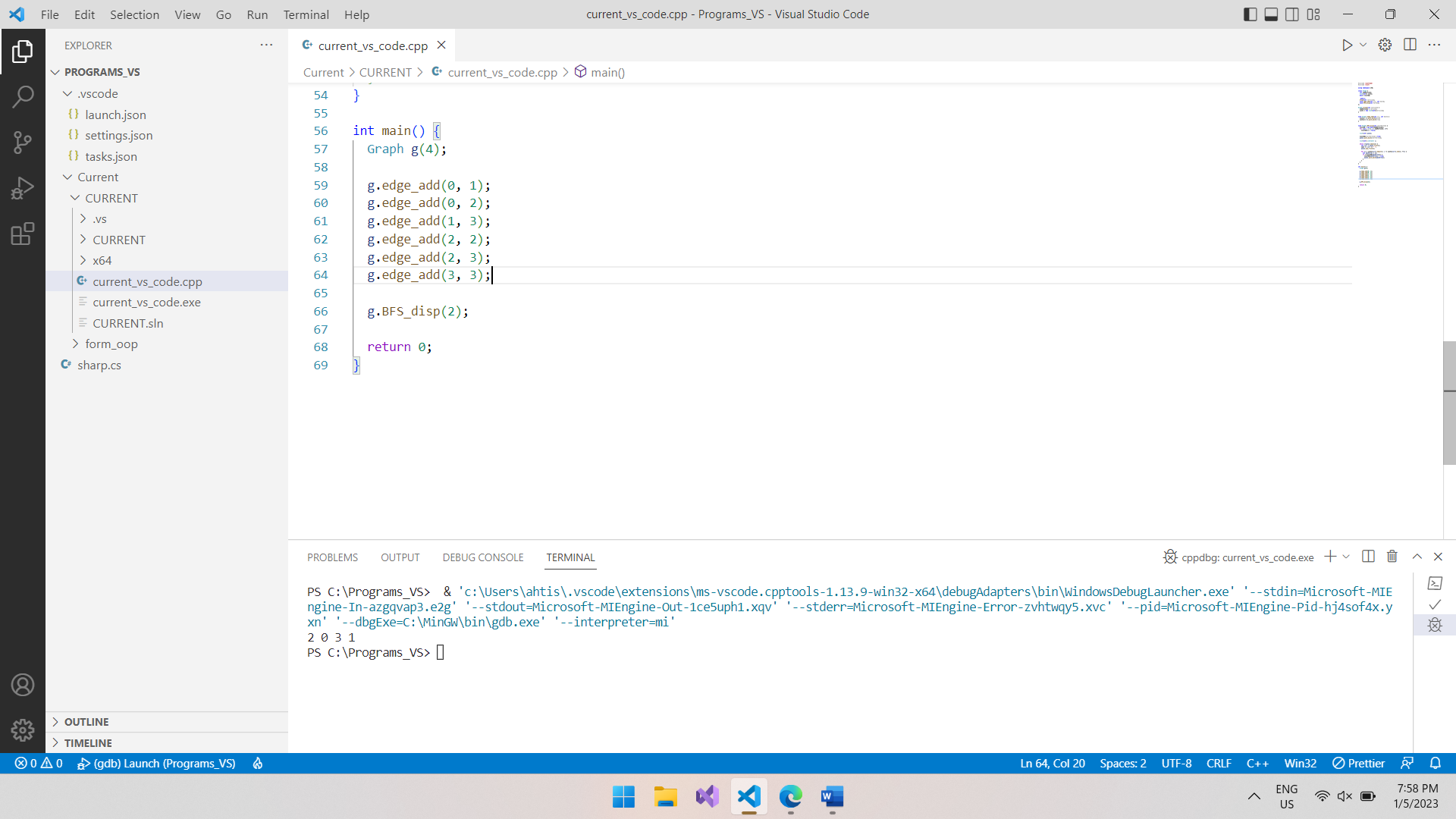
  g.edge\_add(2, 3);

  g.edge\_add(3, 3);

  g.BFS\_disp(2);

  return 0;

}



2. Write a program to traverse Graph through DFS using classes in C++

#include <iostream>

#include <list>

#include <map>

using namespace std;

class Graph {

public:

  map<int, bool> visited;

  map<int, list<int> > adj;

  void addEdge(int v, int w);

  void DFS(int v);

};

void Graph::addEdge(int v, int w)

{

  adj[v].push\_back(w); // Add w to v’s list.

}

void Graph::DFS(int v)

{

  visited[v] = true;

  cout << v << " ";

  list<int>::iterator i;

  for (i = adj[v].begin(); i != adj[v].end(); ++i)

    if (!visited[\*i])

      DFS(\*i);

}

int main()

{

  Graph g;

  g.addEdge(0, 1);

  g.addEdge(0, 2);

  g.addEdge(1, 2);

  g.addEdge(2, 0);

  g.addEdge(2, 3);

  g.addEdge(3, 3);

  g.DFS(0);

  return 0;

}

